REMARKS

Claims 28-39 have been amended. No claims have been added or canceled. Therefore, claims 1-39 remain pending in the application. Reconsideration is respectfully requested in view of the following remarks.

Finality of the Action:

The present Action has been improperly made final. The Examiner states that the new grounds of rejection were necessitated by Applicant's amendment. However, the previous amendments made to independent claims 1 and 14 were non-substantive amendments to improve the consistency of terminology within the claim. The amendments were not made in regard to the previous rejection. Moreover, the body of claim 27 was not amended at all. Surely the Examiner is not suggesting that the addition of the word "storage" in the preamble of claim 27 necessitated a new prior art rejection. It is clear that none of Applicant's arguments in regard to the previous art rejection relied in any way on the amendments. Therefore, the new ground of rejection was clearly necessitated by the arguments presented in Applicant's previous response, not by an amendment. Accordingly, per MPEP 706.07(a) the finality of the present Action is improper and must be withdrawn.

Section 112, Second Paragraph, Rejection:

The Examiner rejected claims 28-39 under 35 U.S.C. § 112, second paragraph, as indefinite. Applicant traverses the rejection and asserts that these claims were not indefinite since no one of ordinary skill in the art would have any trouble ascertaining their meaning. However, to expedite prosecution, claims 28-39 have been amended. Withdrawal of this rejection is respectfully requested.

Section 102(e) Rejection:

The Examiner rejected claims 1, 3-5, 7, 9, 10, 14, 16-18, 20, 22, 23, 27, 29-31, 33, 35 and 36 under 35 U.S.C. § 102(e) as being anticipated by Leong et al. (U.S. Publication 2003/0182292) (hereinafter "Leong"). Applicants traverse the rejection for at least the following reasons.

Claim 1

Leong fails to teach or suggest a class structure based data object enhancer configured to, in part, generate one or more enhanced classes corresponding to the one or more classes such that the one or more enhanced classes are enhanced to persist the data to be persisted according to the determined persistence structure. The Examiner cites paragraphs [0032] and [0035]-[0038], none of which teach or suggest the specific limitations of claim 1. More specifically, Leong fails to teach or suggest anything at all about generating one or more *classes*, much less generating one or more enhanced classes according to the specific limitations of claim 1. To the contrary, Leong teaches a system configured to create persistent Java Vector *objects* (*see e.g.,* paragraph [0032]). However, one skilled in the art would immediately recognize that creating an *object* is not the same as generating a *class*. To the contrary, one skilled in the art would recognize that an object is a particular instance of a class, not a class itself. In fact, Leong affirms this principle in paragraph [0032]: "The Java application 12 calls a constructor to create 0 . . . (m-1) persistent Java Vector objects 22, where the persistent Java Vector objects 22 are instances of the persistent Java Vector class 16" (emphasis added). Clearly, Leong fails to teach or suggest, in the cited art or elsewhere, anything at all about generating classes, much less generating one or more enhanced classes corresponding to the one or more classes such that the one or more enhanced classes are enhanced to persist the data to be persisted according to the determined persistence structure.

Moreover, claim 1 includes generating one or more enhanced classes that correspond to one or more classes. In other words, claim 1 includes a.) one or more classes and b.) one or more enhanced classes corresponding to the one or more classes. According to the limitations of claim 1, an explicit relationship exists between the two types of classes. **Leong fails to teach or suggest such a relationship between classes.** Instead, Leong teaches only a persistent Java Vector class (*see e.g.,* Fig. 2, item 16) and "other" classes (*see e.g.,* Fig. 2, item 18). However Leong fails to teach or suggest any relationship between the persistent Java Vector class and the "other" classes, much less that the persistent Java Vector class corresponds to any of the "other" classes. Accordingly, Leong fails to teach or suggest *generating* one or more enhanced classes *corresponding to* the one or more classes such that the one or more enhanced classes are enhanced to persist the data to be persisted according to the persistence structure.

Furthermore, Leong fails to teach or suggest a class structure based data object enhancer configured to, in part, analyze the *structure* of the one or more *classes* to determine a persistence structure for data of the one or more classes to be persisted. The Examiner cites paragraphs [0032] and [0036], none of which teach or suggest the specific limitations of claim 1. More specifically, Leong fails to teach or suggest analyzing the *structure* of one or more *classes*, much less analyzing the structure of the one or more classes to determine a persistence structure for data of the one or more classes to be persisted. Instead, Leong teaches determining if a particular *call* of a constructor for an *object* "has a pathname in the parameter", as described in paragraphs [0032] and [0036] reproduced below.

> FIG. 3 illustrates the file system data structures that store the persistent data of persistent Java Vector objects in accordance with certain implementations of the invention. The data structures illustrated in FIG. 3 may reside in the computer 2. The Java application 12 calls a constructor to create 0 . . . (m-1) persistent Java Vector objects 22, where the persistent Java Vector objects 22 are instances of the persistent Java Vector class 16. An i^{th} persistent Java Vector object 23 (where i=0, . . . ,m-1) has 0 . . . (n-1) elements 24. All the persistent data of the n elements 24 of the i^{th} persistent Java Vector object 23 (i=0, . . . ,m-1) are kept within a non-volatile memory such as the file system 50, in a data file i 54, and in an index file i 52. Thus for every persistent vector object 0 . . . (m-1) there is an index file i (where i=0, . . . ,m-1) and a data file i (where i=0, . . . ,m-

> 1). The persistent Java Vector objects 22 and the elements 24 may be present in volatile memory. When the Java application 12 stops running, the persistent Java Vector objects 22 and the elements 24 are no longer present. However, the persistent data of the elements 24 of the persistent Java Vector objects 22 are retained for later recovery within the non-volatile file system 50 in the index files and the data files. ([0032], emphasis added)

In paragraph [0032], Leong describes an application calling a constructor to create a persistent object. Leong describes processing this call in paragraph [0036]:

> The Java runtime environment 6 receives (at block 71) from Java application 12 the code that corresponds to the constructor for a Java Vector *object*. The Java runtime environment 6 creates (at block 72) an empty persistent Java Vector object 23. The Java runtime environment 6 determines (at block 74) whether the constructor has a pathname in the parameter. The pathname is a location in persistent storage, such as the filesystem 50 that stores the persistent data, i.e. the index file 52 and the data file 54 of the empty persistent vector object 23. ([0036], emphasis added)

In paragraph [0036], Leong describes a runtime environment determining if the call of the constructor has a pathname in the parameter field. Leong further describes that this pathname indicates where persistent data corresponding to data objects will be stored (*see e.g.,* paragraph [0038]). Leong further teaches, if the pathname is not present in the particular call of the constructor, the runtime environment stores data in the current directory of the runtime environment (*see e.g.,* paragraph [0037]). **Clearly, Leong teaches determining a particular runtime parameter of a particular constructor call. However, determining a particular runtime parameter of a constructor *call* is not the same as analyzing the structure of one or more *classes.***

Applicants respectfully remind the Examiner that anticipation requires the presence in a single prior art reference disclosure of each and every limitation of the claimed invention, arranged as in the claim. M.P.E.P 2131; *Lindemann Maschinenfabrik GmbH v. American Hoist & Derrick Co.*, 221 USPQ 481, 485 (Fed. Cir. 1984). The **identical** invention must be shown in as complete detail as is contained in the claims. *Richardson v. Suzuki Motor Co.*, 9 USPQ2d 1913, 1920 (Fed. Cir. 1989). As discussed

above, Leong clearly fails to disclose the specific limitations of claim 1. Therefore, Leong cannot be said to anticipate claim 1.

Thus, for at least the reasons presented above, the rejection of claim 1 is unsupported by the cited art and removal thereof is respectfully requested. Similar arguments apply to independent claims 14 and 27.

Claim 3

Leong fails to teach or suggest wherein to analyze the structure of the classes, the class structure based enhancer is configured to parse bytecode of the one or more classes to determine class and field attributes. The Examiner cites paragraph [0043], which describes a method for adding an element to a persistent object. However, adding an element to a persistent object is not the same as parsing bytecode of one or more classes to determine class and field attributes. Nowhere does Leong teach or suggest parsing bytecode of one or more classes, much less parsing bytecode of one or more classes to determine class and field attributes.

Thus, for at least the reasons presented above, the rejection of claim 3 is unsupported by the cited art and removal thereof is respectfully requested. Similar arguments apply to claims 16 and 29.

Claim 4

Leong fails to teach or suggest wherein the class structure based enhancer is further configured to generate metadata that includes results of the analysis of the structure of the one or more classes. The Examiner cites paragraphs [0026]-[0028], which include the brief descriptions of Figures 12-13, none of which teach or suggest the specific limitations of claim 4. Furthermore, as described above in regard to claim 1, Leong fails to teach or suggest analyzing the structure of the one or more classes. By

extension, Leong fails to teach or suggest generating metadata that includes results of the analysis of the structure of the one or more classes.

Thus, for at least the reasons presented above, the rejection of claim 4 is unsupported by the cited art and removal thereof is respectfully requested. Similar arguments apply to claims 17 and 30.

Claim 9

Leong fails to teach or suggest wherein to determine a persistence structure for the data of the one or more classes the class structure based enhancer is configured to apply one or more rules to the results of *Java reflection calls to* or *byte code parsing of* the one or more input classes. The Examiner cites paragraphs [0037]-[0039] of Leong, which describe determining a pathname of a particular call of a constructor and storing data according to the pathname, as described above in regard to claim 1. However, nowhere does Leong teach or suggest anything about *Java reflection calls to* or *byte code parsing of* one or more input classes, much less *applying one or more rules* to the results of Java reflection calls to or byte code parsing of the one or more input classes.

Thus, for at least the reasons presented above, the rejection of claim 9 is unsupported by the cited art and removal thereof is respectfully requested. Similar arguments apply to claims 22 and 35.

Section 103(a) Rejection:

The Examiner rejected claims 2, 6, 8, 11, 13, 15, 19, 21, 22, 25, 26, 28, 32, 34 and 37-39 under 35 U.S.C. § 103(a) as being unpatentable over Leong in view of Vachuska, et al. (U.S. Publication 2004/0044687) (hereinafter "Vachuska"). Applicants traverse the rejection for at least the following reasons. Applicants assert that claims 2, 6, 8, 11, 13,

15, 19, 21, 22, 25, 26, 28, 32, 34 and 37-39 recite further distinctions over the cited art. However, since the rejection has been shown to be unsupported for the independent claims, a further discussion of the dependent claims is not necessary at this time.

Furthermore, in regard to claim 12, applicant notes the Examiner rejected claim 12 under 35 U.S.C. § 103(a) as being unpatentable over Leong in view of Vachuska. However, the Examiner's rejection cites the teachings of Chan. Therefore, the rejection of claim 12 is improper. Moreover, neither Leong nor Vachuska, taken singly or in combination, teach or suggest wherein the one or more classes are comprised in a Java ARchive (JAR) file. Similar remarks apply to claims 25 and 38.

CONCLUSION

Applicants submit the application is in condition for allowance, and prompt notice to that effect is respectfully requested.

If any fees are due, the Commissioner is authorized to charge said fees to Meyertons, Hood, Kivlin, Kowert, & Goetzel, P.C. Deposit Account No. 501505/5681-72300/RCK.

Respectfully submitted,

/Robert C. Kowert/
Robert C. Kowert, Reg. #39,255
Attorney for Applicant

Meyertons, Hood, Kivlin, Kowert, & Goetzel, P.C.
P.O. Box 398
Austin, TX 78767-0398
Phone: (512) 853-8850

Date: May 1, 2007